Tarea Evaluable Docker - 2EV_Ejercicio_3_Imagen_con_Dockerfile

Módulo: Despliegue de aplicaciones Web - Distancia

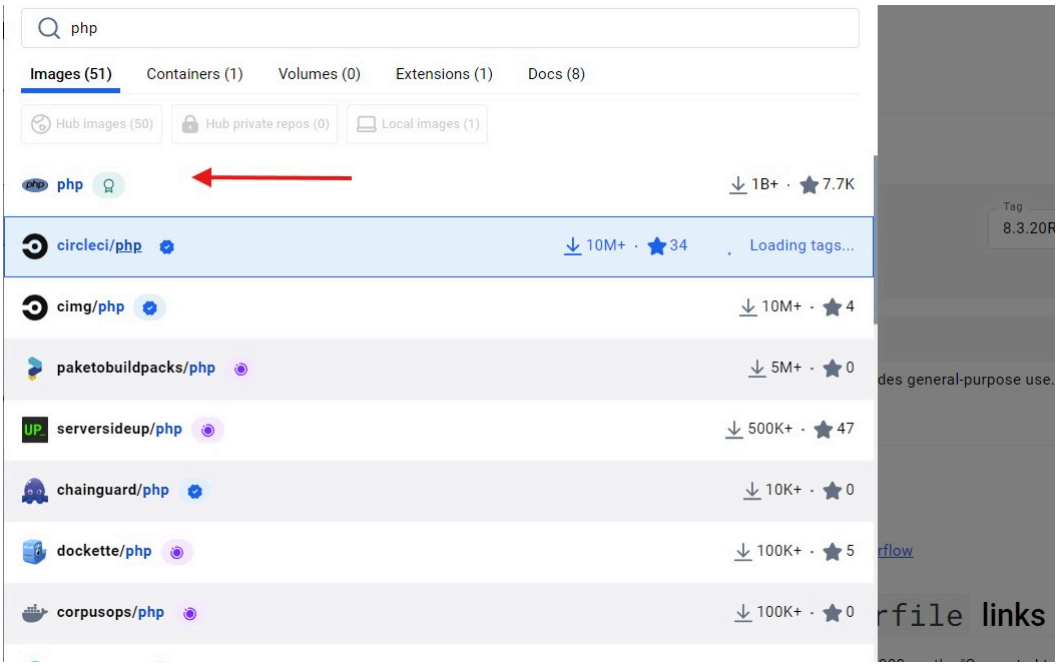
Alumno: Alfonso Dapena Cores

3.1 Consideraciones previas

- El archivo `Dockerfile` es un archivo de texto que contiene las instrucciones precisas para crear una imagen de Docker. En este fichero se define el entorno, las dependencias y configuraciones necesarias para que los contenedores funcionen correctamente. Dentro de un fichero `Dockerfile` nos podemos encontrar con varias instrucciones, siendo las mas comunes:
    1. `FROM`: Identifica la imagen base.
    2. `RUN`: Permite ejecutar comandos en el contenedor.
    3. `WORKDIR`: establece el directorio de trabajo para ejecutar las instrucciones que siguen.
    4. `ENV`: Establece variables de entorno para el contenedor.
    5. `COPY`: Copia archivos y directorios desde el sistema host a la imagen de Docker.
    6. `CMD`: Es comando que se ejecuta por defecto al iniciar un contenedor de Docker.
    7. `ENTRYPOINT`: Define el comando y los parámetros que se ejecutan primero cuando se ejecuta el contenedor.
    8. `EXPOSE`: indica los puertos en los que escuchará un contenedor.

3.1 Creación fichero `Dockerfile`.

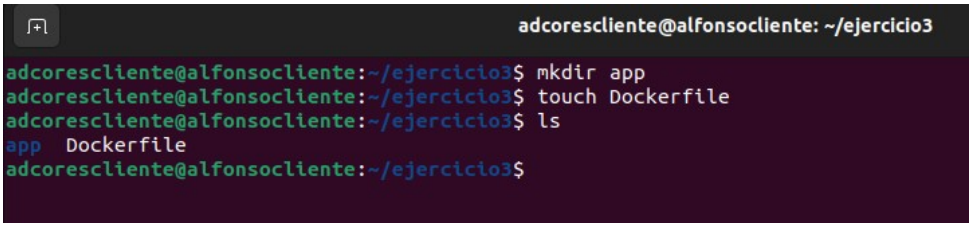
- En primer lugar, antes de preparar el escenario donde vamos a ejecutar nuestro fichero `Dockerfile`, debemos explorar la imagen `PHP`, en nuestra aplicación Docker Desktop:



- En la documentación, nos encontraremos un ejemplo del fichero `Dockerfile` que vamos a ejecutar.

- A continuación, debemos de preparar el entorno donde ejecutaremos nuestro fichero `Dockerfile`. Para ello crearemos los directorios y archivos necesarios para nuestra aplicación con las siguientes líneas de comandos:

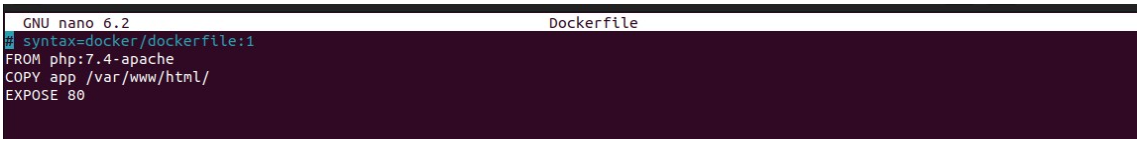
```
$ mkdir app #creamos el directorio que incluye los ficheros que servirá la aplicación
$ touch Dockerfile #creamos el fichero Dockerfile
```



```
adcorescliente@alfonsocliente:~/ejercicio3/app$ ls
estilo.css  fecha.php  index.html
adcorescliente@alfonsocliente:~/ejercicio3/app$
```

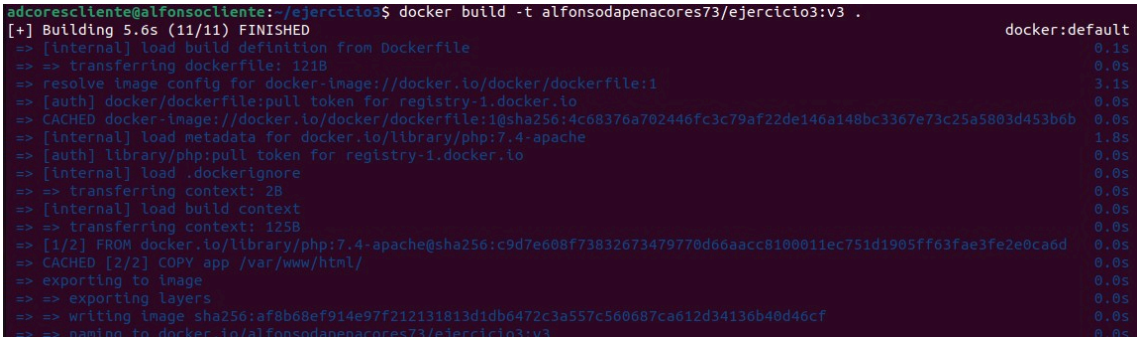
- El siguiente paso consistiría en editar el fichero `Dockerfile` y añadir las instrucciones que se ejecutarán cuando construyamos nuestra imagen.

```
#syntax=docker/dockerfile:1
FROM php:7.4-apache #selecionamos la imagen base con la que construiremos la nuestra
COPY app /var/www/html #copiamos los archivos del directorio app al root de Apache
EXPOSE 80 #nos indica el puerto a través del que se escucharan las peticiones
```



- Una vez creado nuestro fichero `Dockerfile`, procedemos a construir nuestra imagen. Para que podamos subirla con posterioridad a `Docker Hub`, indicaremos nuestro usuario en `Docker Hub`, alfonsodapenacores73, seguido del nombre de la imagen, ejercicio3, y su etiqueta, v3.

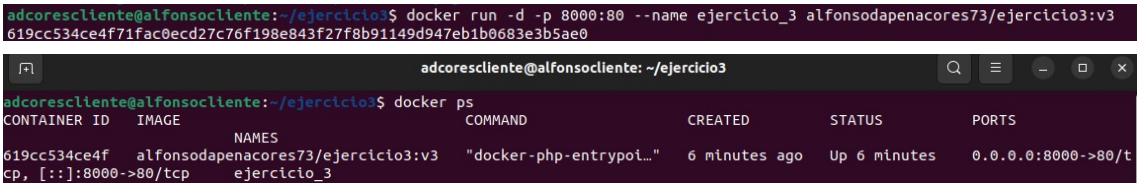
```
$ docker build -t alfonsodapenacores73/ejercicio3:v3
```



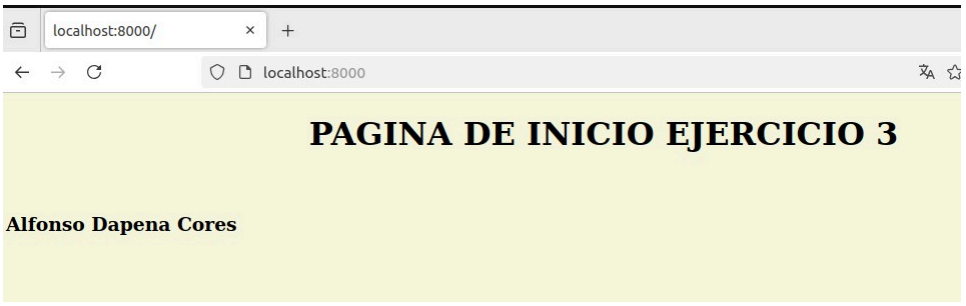
3.3 Creación y ejecución del contenedor.

- Ahora que ya tenemos nuestra imagen lista, procedemos a la creación de nuestro contenedor, al que llamaremos ejercicio_3. A continuación comprobaremos que el contenedor se está ejecutando

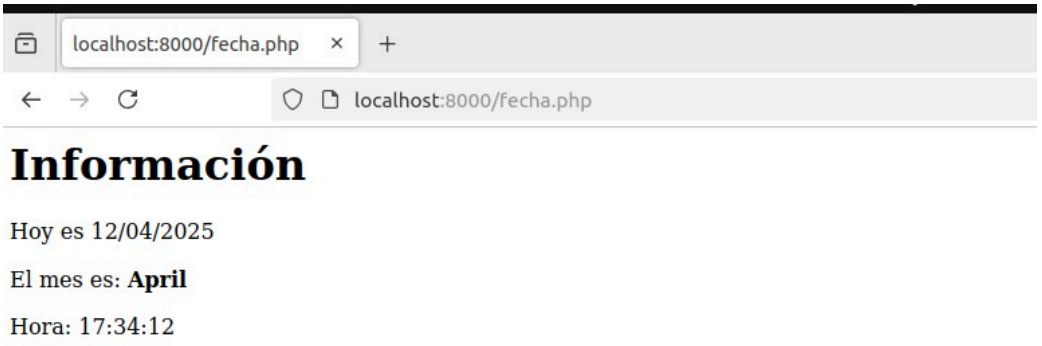
```
$ docker run -d -p 8000:80 --name ejercicio_3
alfonsodapenacores73/ejercicio3:v3
$ docker ps
```



- Lo siguiente será comprobar en nuestro navegador que podemos visualizar la página `index.html` de nuestro servidor apache. Para ello introducimos la siguiente dirección en nuestro navegador: http://localhost:8000



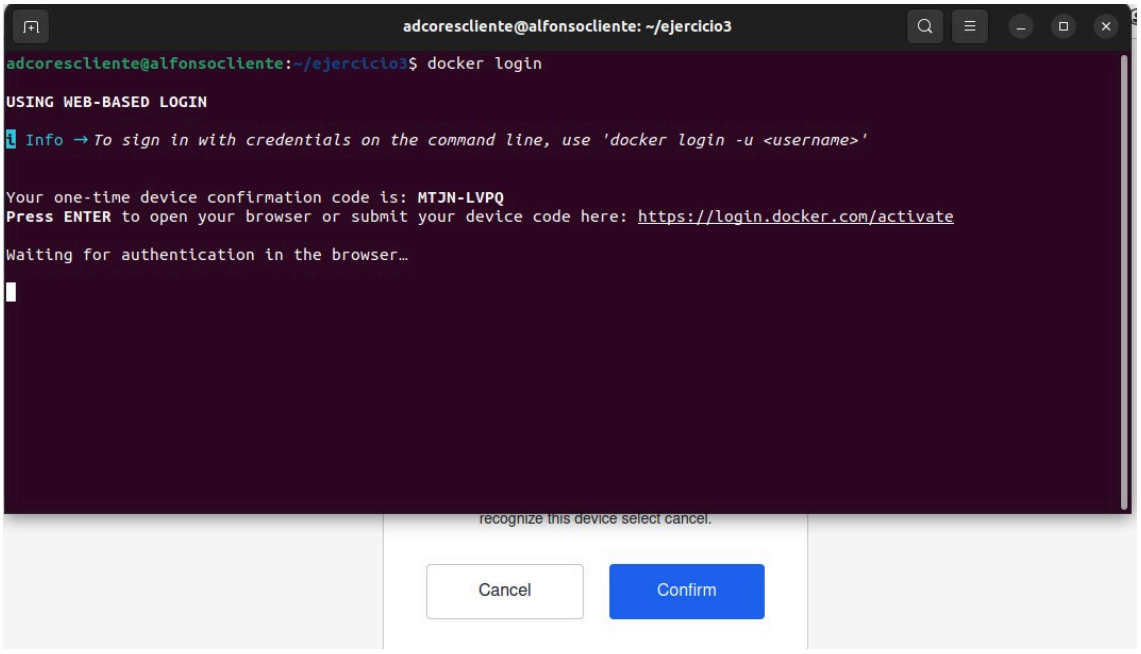
- A continuación comprobamos que el script `php` funciona correctamente introduciendo la siguiente dirección en nuestro navegador : http://localhost:8000/fecha.php

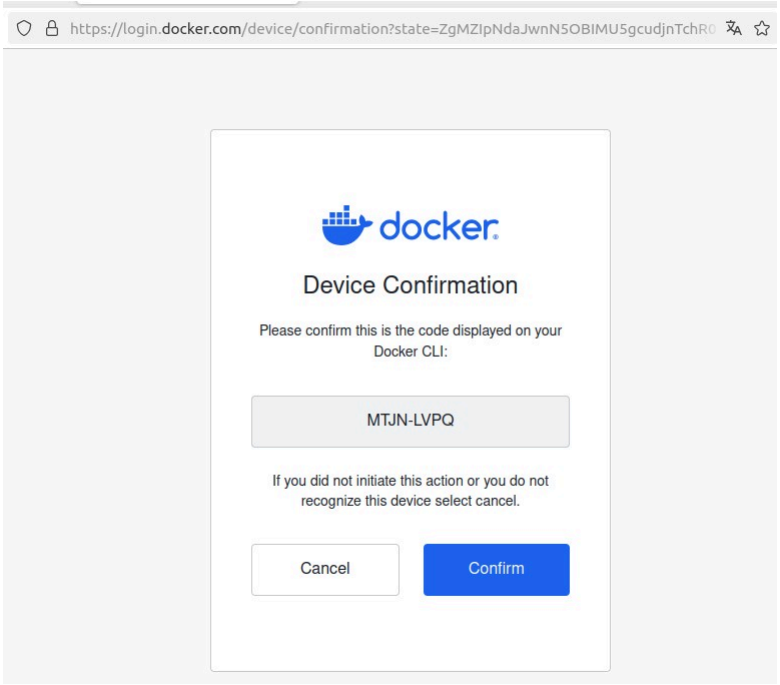


3.4 Subimos la imagen a nuestro repositorio en `Docker Hub`

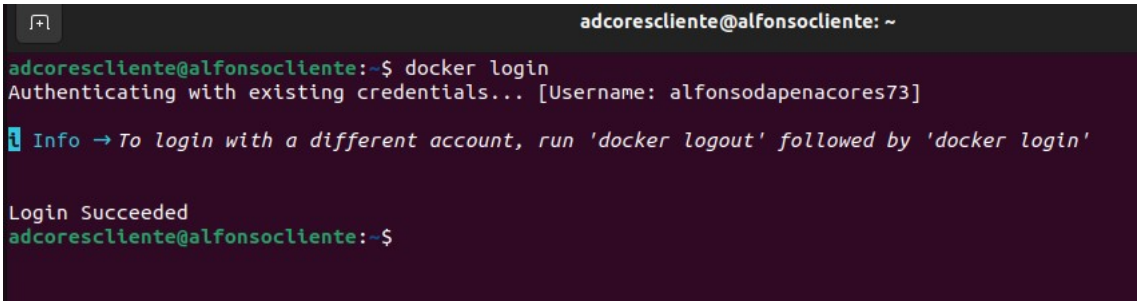
- Para subir nuestra imagen a nuestro repositorio en `Docker Hub`, primero debemos logearnos. Si es la primera vez que lo hacemos, se nos mostrará un enlace que debemos pinchar para autenticarnos:

```
$ docker login
```



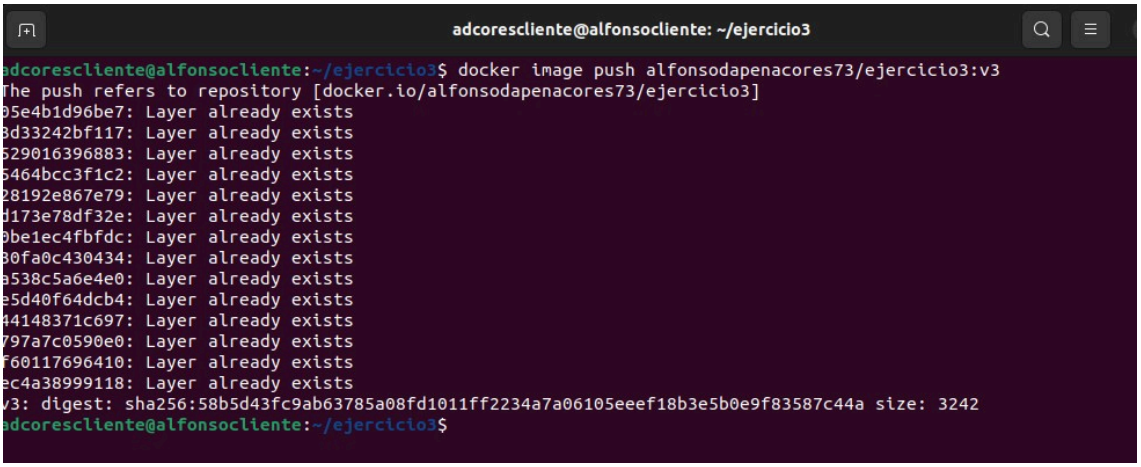


- Después de logearnos con éxito se nos mostrará en pantalla el siguiente mensaje:

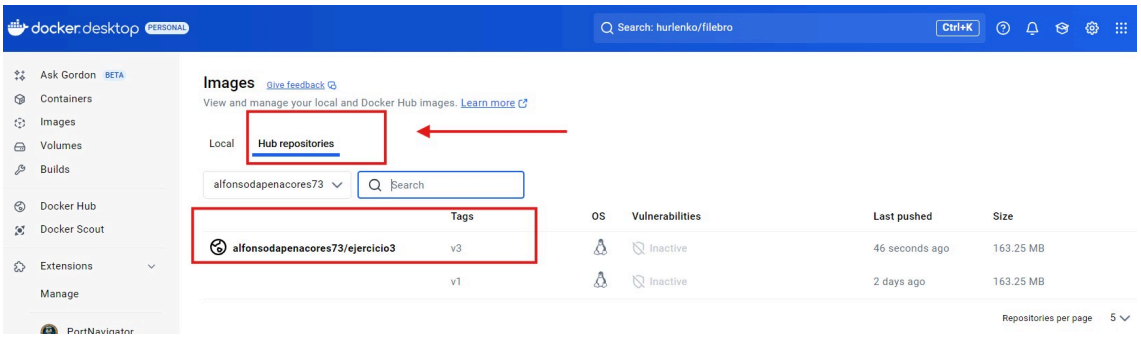


- Ahora que ya nos hemos logeado, deberemos de subir la imagen a nuestro repositorio. Para ello introduciremos el siguiente código:

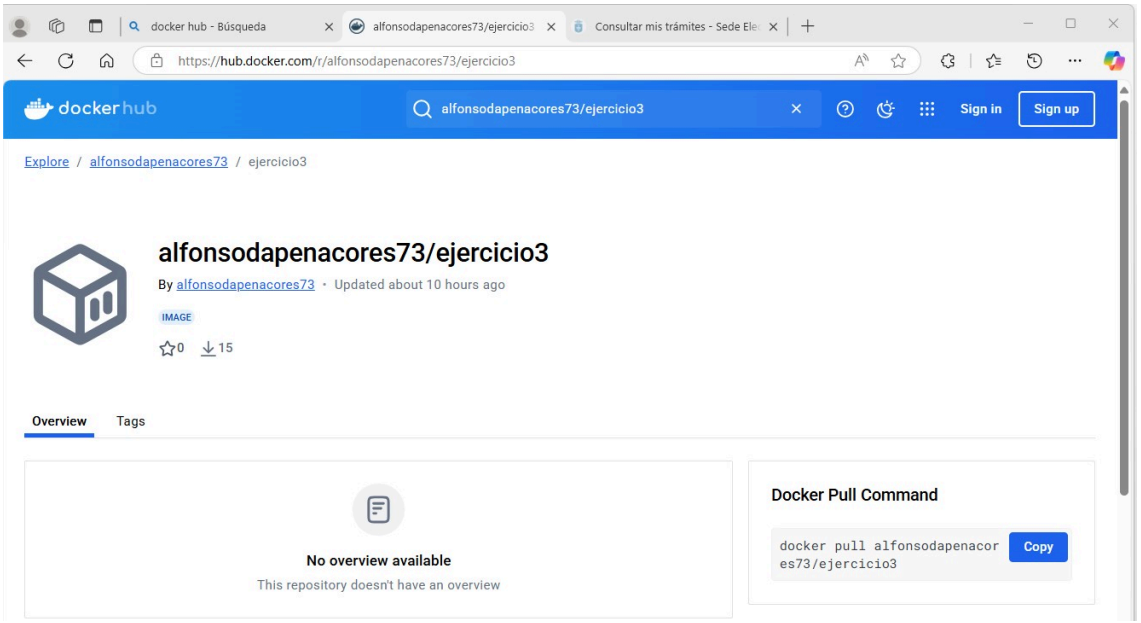
```
$docker image push alfonsodapenacores73/ejercicio3:v3
```



- Accedemos a nuestra aplicación `Docker Desktop` y comprobamos que nuestra imagen se ha subido correctamente:



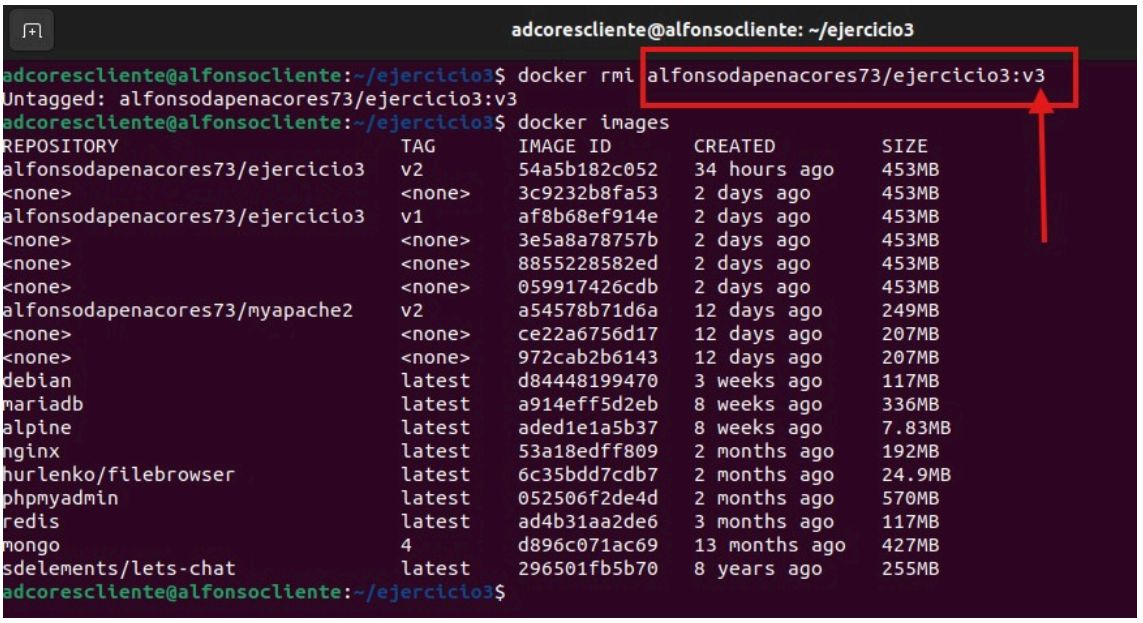
- También podemos comprobar en `Docker Hub`, la existencia de la imagen:



3.5 Borramos la imagen

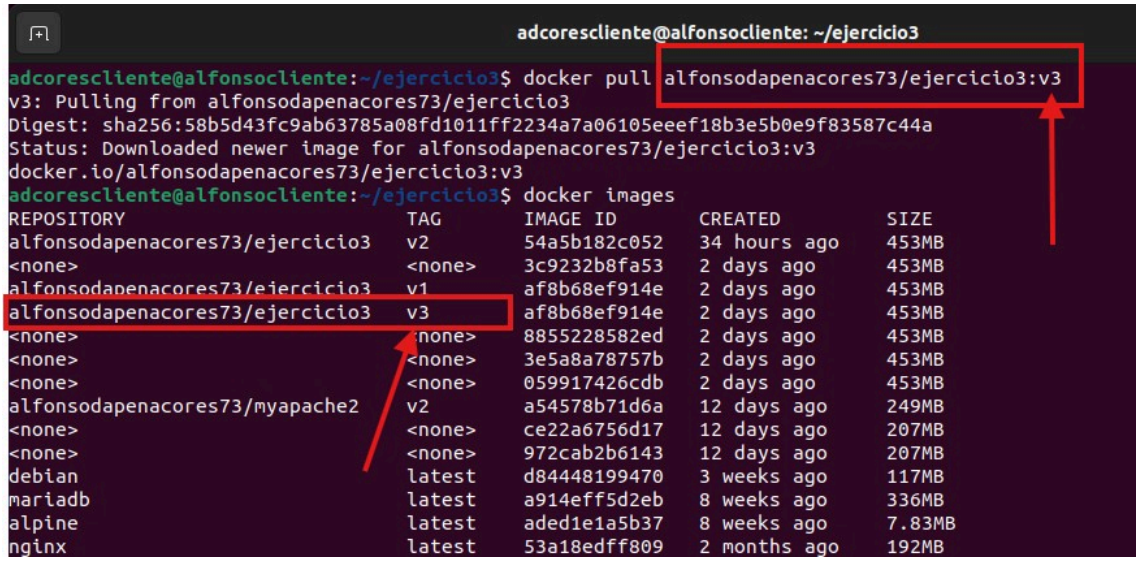
- Una vez hemos comprobado que nuestro contenedor se ejecuta correctamente y que hemos subido nuestra imagen a nuestro repositorio remoto, procedemos a borrar la misma en nuestro repositorio local:

```
$ docker rmi alfomsodapenacores73/ejercicio3:v3 // borramos la imagen
$ docker images // comprobamos que la misma no figure en nuestro repositorio local
```



- Una vez borrada la imagen, nos la descargamos de nuestro repositorio remoto a nuestro repositorio local:

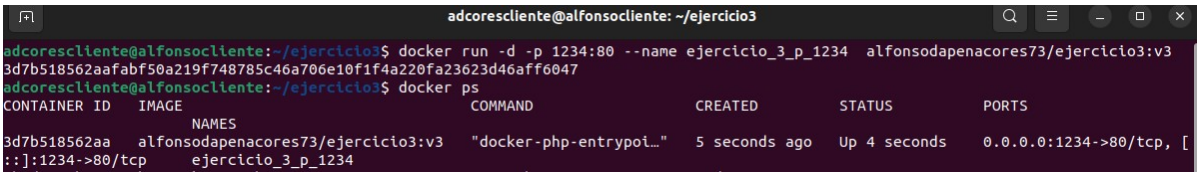
```
$ docker pull alfonsodapenacores73/ejercicio3:v3
```



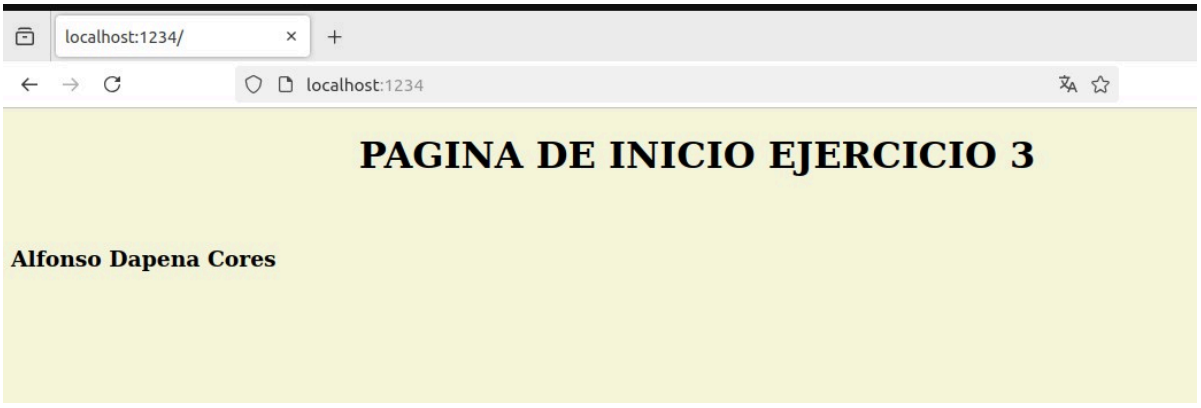
3.6 Creación nuevo contenedor modificando el puerto

- Ahora que hemos descargado la imagen de nuestro repositorio remoto, procedemos a la creación de un nuevo contenedor usando la imagen descargada, pero variando el puerto al 1234. Después, comprobamos que el contenedor se está ejecutando:

```
$docker run -d -p 1234:80 --name ejercicio_3_p_1234
alfonsodapenacores73/ejercicio3:v3
$docker ps
```



- A continuación, abriendo nuestro navegador, comprobamos que podemos acceder al fichero `index.html` de nuestro servidor apache en el puerto 1234, introduciendo la siguiente dirección: http://localhost:1234



- También comprobaremos que nuestro script `php` funciona, introduciendo la dirección http://localhost:1234/fecha.php